John R Bruhling
02-19-2023 19:50
autotunafish@yahoo.com

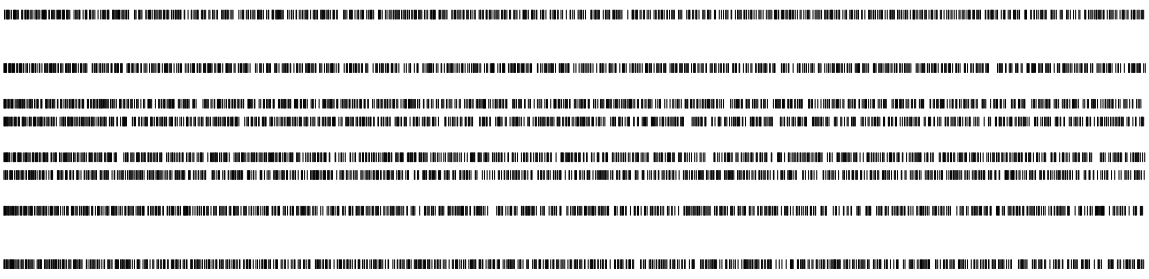
This is a PoC of a practical method of semiprime factorization utilizing insights gained from the Bruhling Sieve and ‘fastcounter program’ which demonstrates the conjecture as a prime-finder and counter. https://archive.org/details/BruhlingSieve
The Bruhling Sieve is a conjecture applying to all primes >5. The premise of the conjecture is as follows:

"Any prime number applied a MOD function with the value 30 will result in one of 8 values between 1 and 30. Those values are 1 | 7 | 11 | 13 | 17 | 19 | 23 | 29. Any prime number (>5) that does not yield one of the 8 values listed from a MOD 30 function applied cannot possibly be prime.

The omitted values 2, 3, and 5 are technically prime but only appear once as MOD values and for the coresponding integers 2, 3, and 5 while the others are repeatedly called. These three values are totally exclusive to themselves while all of the other MOD values are shared amongst all the other primes and it was with this in mind that 8 was decided upon as the number of MOD values given in the description despite the minor technicalities."

This notion is derived from a turtle graphics project which resulted in an image called the 96 Paths of Primality and can be viewed here https://archive.org/details/ulamlogspiral
The image shows an archimedian spiral of length 360 per loop wher the actual pixel length of each single line grows relatively to mimic a relatively increasing velocity (first loop 1-360, second 361-720, third 721-1080..repeat). Along the spiral are plotted the primes as they occured and very clearly can be seen, perhaps not a discernable order of occurances but certainly a type of periodicity in the locations relative to the number line which in that case was base 360 and the 'line' as a circle (I like to think of it all as a cylinder shape). As stated the spiral mirrors itself and it suffices to use a base 90 (MOD 90), and further (MOD30) to get the correct result.

The image was later rendered in straight lines of a count of 30. Beginning in the top left at 1 and moving downward counting to ..=30. The turtle is then reset to origin at a slight offset on the x-axis (to the right) and repeats (31-60, 61-90…). Again, the primes are plotted and in this case from 0..300000.



This eliminates any potentially prime, odd numbers that return 3, 9, 21, or 27 when applied a mod30 function from needing to be checked for primality and reduces the absolute total of those potential prime numbers by 33%.

Furthermore, this gives an interesting insight into prime multiplication as, in a very simplified sense, all primes are essentially one of those eight base numbers plus some increments of 30 and multiplying them is the same as getting the product these base numbers plus the product of their increments of 30's.

For the eight bases numbers there are 36 products between them (8!).

	1,	7,	11,	13,	17,	19,	23,	29
1	1,	7,	11,	13,	17,	19,	23,	29
7		49,	77,	91,	119,	133,	161,	203
11			121,	143,	187,	209,	253,	319
13				169,	221,	247,	299,	377
17					289,	323,	391,	493
19						361,	431,	551
23							529,	667
29								841

If a base-number product is subtracted from the semiprime and a mod 30 function returns 0 then it can be deduced that the prime factors of the semiprime to be possibly within a particular pair of mod paths. The 36 products, when applied the mod 30 function, overlap in such a way (indicated by the coloring) that the path pairing possibilities are exactly SIX for the mod paths 1 and 19 and FOUR for 7, 11, 13, 17, 23 and 29 (Simply applying the mod 30 to the semiprime without subtracting anything is enough to designate it's pairings).

This has the effect of reducing the potential work necessary to factor a semiprime TO between ~11% and ~16% the absolute time of what it may have constituted with no other discriminating search algorithm in place.

The primes specific to a mod path were extracted with the fast counter program by simply printing the prime value that matched to the selected mod path value. These list are set as vectors and used to reference the potential factors. They are very large containing all the realtive primes in the range 0 to 3M and as such the original PoC program become 4 PoC programs with each of them handling 2 of the 8 total mod path pairing groups due to memory constraints applicable to how it is written which aims for speed. These programs can be run independent of each other or simultaneously inside four seperate shells for preference.

The idea of the PoC's is to show that for any given semiprime that is the product of any of the listed primes (in 0-3M) then only one program will match on the base mod and begin working and return the

solution while the 3 others immediately stop. Measured using a millisecond counter, the 3 that stop will return 0ms while the other returns in ~30ms typically on the machine it was tested on (4 thread, 1.5GHZ).